DOCUMENT RESUME

ED 414 932 IR 056 790

AUTHOR	Hockey, Susan
TITLE	Making Technology Work for Scholarship: Investing in the Data.
PUB DATE	1997-04-00
NOTE	16p.; Paper presented at the Conference on Scholarly Communication and Technology (Atlanta, GA, April 24-25, 1997), see IR 056 774.
AVAILABLE FROM	Association of Research Libraries (ARL) Web site: http://www.arl.org/scomm/scat/
PUB TYPE	Reports - Evaluative (142) -- Speeches/Meeting Papers (150)
EDRS PRICE	MF01/PC01 Plus Postage.
DESCRIPTORS	Electronic Libraries; *Electronic Publishing; Electronic Text; Higher Education; *Humanities; Information Technology; Programming Languages; Text Structure; Users (Information)
IDENTIFIERS	Electronic Resources; HTML; *SGML

ABSTRACT

This paper examines issues related to how providers and consumers can make the best use of electronic information, focusing on the humanities. Topics include: new technology or old; electronic text and data formats; Standard Generalized Markup Language (SGML); text encoding initiative; encoded archival description (EAD); other applications of SGML; the relationship between SGML, HTML (HyperText Markup Language) and XML (Extensible Markup Language); SGML and new models of scholarship; and making SGML work effectively. Long before digital libraries became popular, live electronic text was being created for many different purposes, most often, with word processing or typesetting programs. Other electronic texts were created for the purposes of retrieval and analysis. Another commonly used method of storing and retrieving information is a relational database, in which data is assumed to take the form of one or more tables consisting of rows and columns. SGML was designed as a general purpose markup scheme that can be applied to any electronic information. In SGML terms, objects within a document are called elements; the syntax allows the document designer to specify all the possible elements as a Document Type Declaration (DTD) which is a kind of formal model of document structure. The formal structure of SGML means that the encoding of a document can be validated automatically, a process known as parsing. The humanities computing community was among the early adopters of SGML. Following a planning meeting at which representatives of leading humanities computing projects were present, a major international project called the Text Encoding Initiative (TEI) was launched. The TEI SGML application is built on the assumption that all text share some common core of features to which can be added tags for specific application areas. Another SGML application which has attracted a lot of attention in the scholarly community and archival world is the Encoded Archival Description (EAD). Attention must now turn to making SGML work more effectively. (AEF)

****
* Reproductions supplied by EDRS are the best that can be made *
* from the original document. *
****

ED 414 932

Scholarly Communication and Technology

Conference Organized by The Andrew W. Mellon Foundation

at Emory University
April 24-25, 1997

Copyright © of the papers on this site are held by the individual authors or The Andrew W. Mellon Foundation. Permission is granted to reproduce and distribute copies of these works for nonprofit educational or library purposes, provided that the author, source, and copyright notice are included on each copy. For commercial use, please contact **Richard Ekman** at the The Andrew W. Mellon Foundation.

Session #5 Technical Choices and Standards

Making Technology Work for Scholarship: Investing in the Data

Susan Hockey
Department of English
University of Alberta

. . . . . .

Making Technology Work for Scholarship: Investing in the Data

Susan Hockey
Department of English
3-5 Humanities Centre
University of Alberta
Edmonton
Alberta
T6G 2E5
Canada
Phone: 403.492.1029
Fax: 403.492.8142
E-mail: Susan.Hockey@UAlberta.ca

PERMISSION TO REPRODUCE THIS MATERIAL HAS BEEN GRANTED BY
Richard Ekman
TO THE EDUCATIONAL RESOURCES INFORMATION CENTER (ERIC)

U.S. DEPARTMENT OF EDUCATION
Office of Educational Research and Improvement
EDUCATIONAL RESOURCES INFORMATION CENTER (ERIC)
☐ This document has been reproduced as received from the person or organization originating it.
☐ Minor changes have been made to improve reproduction quality.
• Points of view or opinions stated in this document do not necessarily represent official OERI position or policy.

IR056790

BEST COPY AVAILABLE

Revised May 1997

The introduction of any kind of new technology is often a painful and time-consuming process, at least for those who must incorporate it into their everyday lives. This is particularly true of computing technology where the learning curve can be steep, what is learned changes rapidly and ever more new and exciting things seem to be perpetually on the horizon. How can the providers and consumers of electronic information make the best use of this new medium and ensure that the information they create and use will outlast the current system on which it is used? In this paper we examine some of these issues, concentrating on the humanities where the nature of the information studied by scholars can be almost anything and where the information can be studied for almost any purpose.

Today's computer programs are not sophisticated enough to process raw data sensibly. This situation will remain true until artificial intelligence and natural language processing research has made very much more progress than it has so far. Very early on in my days as a humanities computing specialist I saw a library catalogue which had been typed into the computer without anything to separate the fields in the information. There was no way of knowing what was the author, title, publisher or call number of any of the items. The catalogue could be printed out but the titles could not be searched at all, nor could the items in the catalogue be sorted by author name. Although a human can tell which is the author or title from reading the catalogue, a computer program cannot. Something must be inserted in the data to give the program more information. This is a very simple example of markup or encoding which is needed to make computers work better for us. Since we are so far off having the kind of intelligence we really need in computer programs, we must put that intelligence in the data so that computer programs can be informed by it. The more intelligence there is in our data, the better our programs will perform. But what should that intelligence look like? How can we ensure that we make the right decisions in creating it so that computers can really do what we want? Some scholarly communication and digital library projects are among those which are beginning to provide answers to these questions.

1. New Technology or Old?

That having been said, we see many current technology and digital library projects concentrating on using the new technology as an access mechanism to deliver th e old technology. They assume that the typical scholarly product is an article or monograph and that it will be read in a sequential fashion as indeed we have done for hundreds of years ever since these products began to be produced on paper and bound into physical artefacts such as books. The difference is only that instead of going to the library or bookstore to obtain the object, we access it over the network - and then almost certainly have to print a copy of it in order to read it. Of course there are tremendous savings of time for those who have instant access to the network, can find the material they are looking for easily and have high-speed printers. I want to argue here that delivering the old technology via the new is only a transitory phase and that it must not be viewed as an end in itself. Before we embark on the large-scale compilation of electronic information, we must consider how future scholars might use this information and what are the best ways of ensuring that the information will last beyond the current technology.

The old (print) technology developed into a sophisticated model over a long period of time.[1]

Books consist of pages which are bound up in sequential fashion, delivering the text in a single linear sequence. Page numbers and running heads are used for identification purposes. Books also often include other organizational aids such as tables of contents and back-of-the book indexes which are conventionally placed at the beginning and end of the book. Footnotes, bibliographies, illustrations etc provide additional methods of cross-referencing. A title page provides a convention for identifying the book and its author and publication details. The length of a book is often determined by publishers' costs or requirements, rather than by what the author really wants to say about the subject. Journal articles also exhibit similar characteristics, being also designed for reproduction on pieces of paper. Furthermore, the ease of reading of printed books and journals is determined by their typography which is designed to help the reader by reinforcing what the author wants to say. Conventions of typography (headings, italic, bold etc) make things stand out on the page for the human eye.

When we put information into electronic form, we find that we can do many more things with it than we can with a printed book. We can still read it, though not as well as we can read a printed book. The real advantage of the electronic medium is that we can search and manipulate the information in many different ways. We are no longer dependent on the back-of-the-book index to find things in the information but can search for any word or phrase using retrieval software. We no longer need the whole book to look up one paragraph in it, but can just access the piece of information we need. We can also access several different pieces of information at the same time and make links between them. We can find a bibliographic reference and go immediately to the place to which it points. We can merge different representations of the same material into a coherent whole and we can count instances of features within the information. We can thus begin to think of the material we want as "information objects".[12]

To reinforce the arguments we are making here, we can call electronic images of printed pages "dead text" and use the term "live text" for searchable representations of text.[13] For dead text we can use only those retrieval tools which were designed for finding printed items and even then this information must be added as searchable live text, usually in the form of bibliographic references or tables of contents. Of course most of the dead text produced over the last fifteen or so years began its life as live text in the form of wordprocessor documents. The obvious question is how can the utility of that live text be retained and not be lost for ever.

2. Electronic Text and Data Formats

Long before digital libraries became popular, live electronic text was being created for many different purposes, most often, as we have seen, with word processing or typesetting programs. Unfortunately this kind of live electronic text is normally only searchable by the word processing program which produced it and then only in a very simple way. We have all encountered the problems involved in moving from one word processing program to another. Although some of these problems have been solved in more recent versions of the software, maintaining an electronic document as a word processing file is not a sensible option for the long term, unless the creator of the document is absolutely sure that this document will only ever be needed in the short-term future and only ever for the purposes of word processing by the program that created it. Word processed documents contain typographic markup or codes to specify the formatting. If there was no markup the document would be much more difficult to read. However typesetting markup is ambiguous and thus cannot be used sensibly by any retrieval program. For example, italics can be used for titles of books, or for emphasized words, or for foreign words. With typographic markup we cannot distinguish titles of books from foreign

words, which we may at some stage want to search for separately.

Other electronic texts were created for the purposes of retrieval and analysis. Many such examples exist, ranging from the large text databases of legal statutes to humanities collections such as the Thesaurus Linguae Graecae (TLG) and the Trésor de la langue française. These projects all realized that they needed to put some intelligence into the data in order to search it effectively. Most devised markup schemes which focus on ways of identifying the reference citations for items which have been retrieved, for example in the TLG, the name of the author, work, book and chapter number. They do not provide easily for representing items of interest within a text, for example foreign words or quotations. Most of these markup schemes are specific to one or two computer programs and texts prepared in them are not easily interchangeable. A meeting in 1987 examined the very many markup schemes for humanities electronic texts and concluded that the present situation was "chaos".[4] No existing markup scheme satisfied the needs of all users and much time was being wasted converting from one deficient scheme to another.

Another commonly used method of storing and retrieving information is a relational database, as, for example, in Microsoft Access or dBASE, or the mainframe program Oracle. In it, data is assumed to take the form of one or more tables consisting of rows and columns, that is rectangular structures.[5] A simple table of biographical information may have rows representing people and columns holding information about those people, for example, name, date of birth, occupation etc. When a person has more than one occupation, the data becomes clumsy and the information is best represented in two tables where the second has a row for each person's occupation. The tables are linked or related by the person. A third table may hold information about the occupations. It is not difficult for a human to conceptualize the data structures of a relational database or for a computer to process them. Relational databases work well for some kinds of information, for example address lists etc, but in reality not much data in the real world fits well into rectangular structures. This means that the information is distorted when it is entered into the computer, and processing and analyses are carried out on the distorted forms, whose distortion tends to be forgotten. Relational databases also force the allocation of information to fixed data categories, whereas, in the humanities at any rate, much of the information is subject to scholarly debate and dispute, requiring multiple views of the material to be represented. Furthermore, getting information out of a relational database for use by other programs usually requires some programming knowledge.

The progress of too many retrieval and database projects can be characterized as follows. The project decided that it wants to "make a CD-ROM". It finds that it has to investigate possible software programs for delivery of the results and chooses the one which has the most seductive user interface or most persuasive salesperson. If the data includes some non-standard characters, being able to display them on the screen is considered the highest priority and the functions that are needed to manipulate those characters are not looked at very hard. Data is then entered directly into this software over a period of time during which the software interface begins to look outmoded as technology changes. By the time that the project has finished entering the data, the software company has gone out of business leaving the project with a lot of valuable information in a proprietary software format which is no longer supported. More often than not the data is lost and much time and money has been wasted. The investment is clearly in the data and it makes sense to ensure that this is not dependent on one particular program, but can be used by other programs as well.

3. The Standard Generalized Markup Language (SGML)

Given the time and effort involved in creating electronic information, it makes sense to step back and think about how to ensure that the information can outlast the computer system on which it is created, and can also be used for many different purposes. These are the two main principles of the Standard Generalized Markup Language (SGML) which became an international standard (ISO 8879) in 1986.[6] SGML was designed as a general purpose markup scheme that can be applied to many different types of documents and in fact to any electronic information. It consists of plain ASCII files which can easily be moved from one computer system to another. SGML is a descriptive language. Most encoding schemes prior to SGML use prescriptive markup. One example of prescriptive markup is word processing or typesetting codes embedded in a text which give instructions to the computer such as "center the next line" or "print these words in italic". Another is fielded data which is specific to a retrieval program, for example, reference citations or author's names which must be in a specific format for the retrieval program to recognize them as such. By contrast, a descriptive markup language merely identifies what the components of a document are. It does not give specific instructions to any program. In it, for example, a title is encoded as a title, or a paragraph as a paragraph. This very simple approach ultimately allows much more flexibility. A printing program can print all the titles in italic. A retrieval program can search on the titles and a hypertext program can link to and from the titles, all without making any changes to the data.

Strictly, SGML itself is not a markup scheme, but a kind of computer language for defining markup or encoding schemes. SGML markup schemes assume that each document consists of a collection of objects which nest within each other or are related to each other in some other way. These objects or features can be almost anything. Typically they are structural components such as title, chapter, paragraph, heading, act, scene, speech, but they can also be interpretive information such as parts of speech, names of people and places, quotations (direct and indirect) and even literary or historical interpretation. The first stage of any SGML-based project is document analysis where the project identifies all the textual features which are of interest and the relationships between them. This can take some time, but it is worth investing the time since a thorough document analysis can ensure that data entry proceeds smoothly and that the documents are easily processable by computer programs.

In SGML terms, the objects within a document are called elements. They are identified by a start and end tag as follows: <title>Pride and Prejudice</title>. The SGML syntax allows the document designer to specify all the possible elements as a Document Type Declaration (DTD) which is a kind of formal model of the document structure. The DTD indicates which elements are contained within other elements, which are optional, which can be repeated etc. For example, in simple terms a journal article consists of a title, one or more authors, an optional abstract, an optional list of keywords, followed by the body of the article. The body may contain sections, each beginning with a heading followed by one or more paragraphs of text. The article may finish with a bibliography. The paragraphs of text may contain other features of interest including quotations, lists, names, as well as links to notes. A play has a rather different structure of which an outline could be: title, author, castlist, one or more acts each containing one or more scenes, each containing one or more speeches and stage directions etc.

SGML elements may also have attributes which further specify or modify the element. One use of attributes may be to normalize the spelling of names for indexing purposes. For example, the name Jack Smyth could be encoded as <name norm="SmithJ"> Jack Smyth</name>, but indexed under S as if it were Smith. Attributes can also be used to normalize date forms for

sorting, for example <date norm=19970315>the Ides of March 1997</date>. Another important function of attributes is to assign a unique identifier to each instance of each SGML element within a document. This can be used as a cross-reference by any kind of hypertext program. The list of possible attributes for an element may be defined as a closed set, allowing the encoder to pick from a list, or it may be entirely open.

SGML has another very useful feature. Any piece of information can be given a name and be referred to by that name in an SGML document. These are called entities and are enclosed in & and ;. One use is for non-standard characters, where for example é can be encoded as é thus ensuring that it can be transmitted easily across networks and from one machine to another. A standard list of these characters exists, but the document encoder can also create more. Entity references can also be used for any boilerplate text. This avoids repetitive typing of words and phrases which are repeated, thus also reducing the chance of errors. An entity reference can be resolved to any amount of text from a single letter up to something like a whole chapter.

The formal structure of SGML means that the encoding of a document can be validated automatically, a process known as parsing. The parser makes use of the SGML DTD to determine the structure of the document and can thus help to eliminate whole classes of encoding errors, before the document is processed by an application program. For example, an error can be detected if the DTD specifies that a journal article must have one or more authors, but the author's name has been omitted accidentally. Mistyped element names can be detected as errors as can elements which are wrongly nested, for example, an act within a scene when the DTD specifies that acts contain scenes. Attributes can also be validated when there is a closed set of possible values. The validation process can also detect un-resolved cross-references which use SGML's inbuilt identifiers. The SGML document structure and validation process means that any application program can operate more efficiently since it derives information from the DTD about what to expect in the document. It follows that the stricter the DTD, the easier it is to process the document. However very strict DTDs may force the document encoder to make decisions which simplify what is being encoded. Free DTDs might better reflect the nature of the information but usually require more processing. Another advantage of SGML is very apparent here. Once a project is underway, if a document encoder finds a new feature of interest, that feature can simply be added to the DTD without the need to restructure work that has already been done. Of course many documents can be encoded and processed with the same DTD.

4. The Text Encoding Initiative

The humanities computing community was among the early adopters of SGML, for two very simple reasons. Humanities primary source texts can be very complex, and they need to be shared and used by different scholars. They can be in different languages and writing systems and can contain textual variants, non-standard characters, annotations and emendations, multiple parallel texts, hypertext links, as well as having complex canonical reference systems. In electronic form, these texts can be used for many different purposes including the preparation of new editions, word and phrase searches, stylistic analyses and research on syntax and other linguistic features. By 1987 it was clear that many encoding schemes existed for humanities electronic texts, but none was sufficiently powerful to allow for all the different features which might be of interest. Following a planning meeting at which representatives of leading humanities computing projects were present, a major international project called the Text Encoding Initiative (TEI),was launched.[7] Sponsored by the Association for Computers and the

Humanities, the Association for Computational Linguistics and the Association for Literary and Linguistic Computing, the TEI enlisted the help of volunteers all over the world to define what features might be of interest to humanities scholars working with electronic text. It built on the expertise of groups such as the Perseus Project (then at Harvard, now at Tufts University), the Brown University Women Writers Project, the Alfa Informatica Group in Groningen, Netherlands, and others who were already working with SGML, to create SGML tags which could be used for many different types of text.

The TEI published its *Guidelines for the Encoding and Interchange of Electronic Texts*, in May 1994 after over six years' work. The Guidelines identify some four hundred tags, but of course no list of tags can be truly comprehensive and so the TEI builds up its DTDs in a way which makes it easy for users to modify them. The TEI SGML application is built on the assumption that all text share some common core of features to which can be added tags for specific application areas. Very few tags are mandatory and most of these are concerned with documenting the text and will be discussed further below. The TEI Guidelines are simply guidelines. They serve to help the encoder identify features of interest and provide the DTDs with which the encoder will work. The core consists of the header which documents the text, plus basic structural tags and common

features such as lists, abbreviations, bibliographic citations, quotations, simple names and dates etc. The user selects a base tag set of which the following have been defined at present: prose, verse, drama, dictionaries, spoken texts, terminological data. To this are added one or more additional tag sets. The options here include simple analytic mechanisms, linking and hypertext, transcription of primary sources, critical apparatus, names and dates, and some methods of handling graphics. The TEI has also defined a method of handling non-standard alphabets by using a Writing System Declaration which the user specifies. It can also be used for non-alphabetic writing systems, for example, Japanese. Building a TEI DTD has thus been likened to the preparation of a pizza where the base tag set is the base, the core tags are the tomato and cheese and the additional tag sets are the toppings.

One of the issues addressed at the TEI planning meeting was the need for documentation of an electronic text. Many electronic texts now exist about which little is known, either what source text they were taken from, what decisions were made in encoding the text and what changes have been made to the text. All this information is extremely important to a scholar wanting to work on the text, since it will determine the academic credibility of his or her work. Unknown sources are unreliable at best and lead to inferior work. Experience has shown that electronic texts are more likely to contain errors or have bits missing, but these are more difficult to detect than with printed material. It seems that one of the main reasons for this lack of documentation for electronic texts was simply that there was no common methodology for providing it.

The TEI examined various models for documenting electronic texts and concluded that some SGML elements placed as a header at the beginning of an electronic text file would be the most appropriate way of providing this information. Since the header is part of the electronic text file, it is more likely to remain with that file throughout its life. It can also be processed by the same software as the rest of the text. The TEI header contains four major sections.[8] One is a bibliographic description of the electronic text file using SGML elements which map closely on to some MARC fields. The electronic text is a different intellectual object from the source from which it was created and the source is thus also identified in the header. The encoding description section provides information about the principles used in encoding the text, for example whether the spelling has been normalized, treatment of end-of-line hyphens, etc. For

spoken texts the header provides a way of identifying the participants in a conversation and attaching a simple identifier to each of them which can then be used as an attribute on each utterance. The header also provides a revision history of the text indicating who made what changes to it and when.

As far as can be ascertained the TEI header is the first systematic attempt to provide documentation for an electronic text a part of the text file itself. A good many projects are now using it, but experience has shown that it would perhaps benefit from some revision. Scholars find it hard to create good headers. Some elements in the header are very obvious, but the relative importance of the remaining elements is not so clear. At some institutions librarians are creating TEI headers, but they need training in the use and importance of the non-bibliographic sections and in how the header can be used by computer software other the bibliographic tools which they know well.

5. Encoded Archival Description (EAD)

Another SGML application which has attracted a lot of attention in the scholarly community and archival world is the Encoded Archival Description (EAD). First developed by Daniel Pitti at the University of California at Berkeley and now taken over by the Library of Congress, the EAD is an SGML application for archival finding aids.[9] Finding aids are very suitable for SGML because they are basically hierarchic in structure. In simple terms a collection is divided into series which consist of boxes which contain folders etc. Prior to the EAD, there was no effective standard way of preparing finding aids. Typical projects created a collection level record in one of the bibliographic utilities such as RLIN and used their own procedures, often a word processing program, for creating the finding aid. Possibilities now exist for using SGML to link electronic finding aids with electronic representations of the archival material itself. One such experiment, conducted at the Center for Electronic Texts in the Humanities (CETH), has created an EAD-encoded finding aid for part of the Griffis Collection at Rutgers University and encoded a small number of the items in the collection (19th century essays) in the TEI scheme.[10] The user can work with the finding aid to locate the item of interest and then move directly to the encoded text and an image of the text to study the item in more detail. The SGML browser program Panorama allows the two DTDs to exist side by side and in fact uses an extended pointer mechanism devised by the TEI to move from one to the other.

6. Other Applications of SGML

SGML is now being widely adopted in the commercial world as companies see the advantage of investment in data which will move easily from one computer system to another. It is worth noting that the few books on SGML which appeared early in its life where intended for an academic audience. More recent books are intended for a commercial audience and emphasize the cost savings involved in SGML as well as the technical requirements. This is not to say that these books are not of any value to academic users. The SGML Web pages list many projects in the areas of health, legal documents, electronic journals, rail and air transport, semiconductors, the US Internal Revenue Service and more. SGML is extremely useful for technical documentation as can be evidenced by the list of customers on the Web page of one of the major SGML software companies INSO/EBT. This includes United Airlines, Novell, British Telecom, AT&T, Shell, Boeing, Nissan and Volvo.

SGML need not only be used with textual data. It can be used to describe almost anything. SGML should not therefore be seen as an alternative to Acrobat, PostScript or other document formats, but as a way of describing and linking together documents in these and other formats, forming the "underground tunnels" which make the documents work for users.[11] SGML can be used to encode the searchable textual information which must accompany images or other formats in order to make them useful. With SGML the searchable elements can be defined to fit the data exactly and can be used by different systems. This is in contrast with storing image data in some proprietary database system, as often happens. Further down the line we can imagine a situation where a scholar wants to examine the digital image of a manuscript and also have available a searchable text. He or she may well find something of interest on the image and want to go to occurrences of the same feature elsewhere within the text. In order to do this, the encoded version of the text must know what that feature of interest is and where it occurs on the digital image. Knowing which page it is on is not enough. The exact position on the page must be encoded. This information can be represented in SGML which thus provides the sophisticated kind of linking needed for scholarly applications. SGML structures can also point to places within a recording of speech or other sound and can be used to link the sound to a transcription of the conversation, again enabling the sound and text to be studied together. Other programs exist which can perform these functions, but the problem with all of them is that they use a proprietary data format which cannot be used for any other purpose.

7. SGML, HTML and XML

The relationship between SGML and the HyperText Markup Language (HTML) needs to be clearly understood. Although not originally designed as such, HTML is now an SGML application, even though many HTML documents exist which cannot be validated according to the rules of SGML. HTML consists of a set of elements which are interpreted by Web browsers for display purposes. The HTML tags were designed for display and not for other kinds of analysis, which is why only crude searches are possible on Web documents. HTML is a rather curious mixture of elements. Larger ones such as <body>, <h1> etc, <p> for paragraph, <ul> for unordered list are structural, but the smaller elements such as <b> for bold, <i> for italic are typographic, which, as we have seen above, is ambiguous and thus cannot be searched effectively. HTML version 3 attempts to rectify this somewhat by introducing a few semantic level elements, but these are very few in comparison with those identified in the TEI core set. HTML can be a good introduction to structured markup. Since it is so easy to create, many projects begin by using HTML and graduate to SGML once they have got used to working with structured text and begin to see the weakness of HTML for anything other than the display of text. SGML can easily be converted automatically to HTML for delivery on the Web, and Web clients have been written for the major SGML retrieval programs.

The move from HTML to SGML can be substantial and in 1996 work began on XML (Extensible Markup Language) which is a simplified version of SGML for delivery on the Web. It is "an extremely simple dialect of SGML" the goal of which "is to enable generic SGML to be served, received, and processed on the Web in the way that is now possible with HTML". XML is being developed under the auspices of the World Wide Web Consortium and the first draft of the specification for it was available by the SGML conference in December 1996. Essentially it is SGML with some of the more complex and esoteric features removed. It has been designed for interoperability with both SGML and HTML, to fill the gap between the HTML which is too simple and full-blown SGML which can be complicated. As yet there is no specific XML software, but the work of this group has considerable backing and the design of XML has

proceeded quickly.[12]

8. SGML and New Models of Scholarship

SGML's object-like structures make it possible for scholarly communication to be seen as "chunks" of information which can be put together in different ways. Using SGML we no longer have to squeeze the product of our research into a single linear sequence of text, whose size is often determined by the physical medium in which it will appear, but can organize it in many different ways, privileging some for one audience and others for a different audience. Some projects are already exploiting this potential and I am collaborating in two which are indicative of the way I think humanities scholarship will develop in the 21st century. Both make use of SGML to create information objects which can be delivered in many different ways.

The Model Editions Partnership (MEP) is defining a set of models for electronic documentary editions.[13] Directed by David Chesnutt of the University of South Carolina with the TEI Editor, C. Michael Sperberg-McQueen, and myself as co-coordinators, the MEP also includes seven documentary editing projects. Two of these projects are creating image editions and the other five are preparing letterpress publications. These documentary editions provide the basic source material for the study of American history, by adding the historical context which makes the material meaningful to readers. Much of this source material consists of letters which often refer to people and places by words which only the author and recipient understand. A good deal is in handwriting which only scholars specializing the field can read. Documentary editors prepare the material for publication by transcribing the documents, organizing the sources into a coherent sequence which tells the story (the history) behind them, and annotating them with information to help the reader understand them. However, the printed page is not very good vehicle for conveying the information which documentary editors need to say. It forces one organizing principle on the material (the single linear sequence of the book), when the material could well be organized in several different ways (chronologically or by recipient of letters). Notes must appear at the end of an item to which they refer or at the end of the book. When the same note, for example, a short biographical sketch of somebody mentioned in the sources, is needed in several places, it can only appear once and then be cross-referenced by page numbers, often to earlier volumes. If something has been crossed out and rewritten in a source document, this can only represented clumsily in print, even though it may reflect a change of mind which altered the course of history.

At the beginning of the MEP project, the three coordinators visited all seven partner projects, showed them some very simple demonstrations and then invited them to "dream" about what they would like to do in this new medium. The ideas collected during these visits were the incorporated into a prospectus for electronic documentary editions. The MEP sees SGML as the key to providing all the functionality outlined in the prospectus. The MEP has developed an SGML DTD for documentary editions which is based on the TEI and has begun to experiment with delivery of samples from the partner projects. The material for the image editions is wrapped up in an "SGML envelope" which provides the tools to access the images. This envelope can be generated automatically from the relational databases in which the image access information is now stored. For the letterpress editions, many more possibilities are apparent. If desired, it will be possible to merge material from different projects which are working on the same period of history. It will be possible to select subsets of the material easily, by any of the features that are tagged. This means that editions for high school students or the general public could be created almost automatically from the archive of scholarly material. With a click of a

mouse the user can go from a diplomatic edition to a clear reading text and thus trace the author's thoughts as the document was being written. The documentary editions also include very detailed conceptual indexes compiled by the editors. It will be possible to use these as an entry point to the text and also to merge indexes from different projects. The MEP sees the need for making "dead text" image representations of existing published editions available quickly and believes that these can be made much more useful by wrapping them in SGML and using the conceptual indexes as an entry point to them.

The second project is even more ambitious than the MEP, since it is dealing with entirely new material and has been funded for five years. The Orlando Project at the Universities of Alberta and Guelph is a major collaborative research initiative funded by the Canadian Social Sciences and Humanities Research Council.[14] Directed by Patricia Clements, the project is creating an Integrated History of Women's Writing in the British Isles, which will appear in print and electronic formats. The project has a team of graduate research assistants carrying out basic research for the project in libraries and elsewhere. The research material they are assembling is being encoded in SGML so that it can be retrieved in many different ways. SGML DTDs have been designed to reflect the biographical details for each woman writer, also their writing history, other historical events which influenced their writing, a thesaurus of keyword terms etc. The DTDs are based on the TEI but they incorporate much descriptive and interpretive information, reflecting the nature of the research and the views of the literary scholars in the team. Tagsets have been devised for topics such as the discussion of issues of authorship and attribution, for genre issues and for issues of reception of an author's work.

The Orlando Project is thus building up an SGML-encoded database of many different kinds of information about women's writing in the British Isles. The SGML encoding, for example, greatly assists in the preparation of a chronology by allowing the project to pull out all chronology items from the different documents and sort them by their dates. It facilitates an overview of where the women writers lived, their social background, what external factors influenced their writing etc. It helps the creation and consistency of new entries since the researchers can see immediately if similar information has already been encountered. The authors of the print volumes will draw on this SGML archive as they write, but the archive can also be used to create many different hypertext products for research and teaching.

Both Orlando and the MEP are essentially working with pieces of information, which can be linked in many different ways. The linking, or rather the interpretation which gives rise to the linking is essentially what humanities scholarship is about. When the information is stored as encoded pieces of information, it can be put together in many different ways and used for many different purposes of which creating a print publication is only one. We can expect other projects to begin to work in this way as they see the advantages of encoding the features of interest in their material and manipulating them in different ways.

It is useful to look briefly at some other possibilities. Dictionary publishers were among the first to use SGML. (Although not strictly SGML, since it does not have a DTD, the Oxford English Dictionary was the first academic project to use structured markup.) When well designed, the markup enables the dictionary publishers to create spin-off products for different audiences by selecting a subset of the tagged components of an entry. A similar process can be used for other kinds of reference works. Tables of contents, bibliographies, and indexes can all be compiled automatically from SGML markup and can also be cumulative across volumes or collections of material.

The MEP is just one project that uses SGML for scholarly editions. A notable example is the CD-ROM of Chaucer's Wife of Bath's Prologue prepared by Peter Robinson and published by Cambridge University Press in 1996. This CD-ROM contains all fifty-eight pre-1500 manuscripts of the text with encoding for all the variant readings, as well as digitized images of every page of all the manuscripts. Software programs provided with the CD-ROM can manipulate the material in many different ways enabling a scholar to collate manuscripts, move immediately from one manuscript to another, compare transcriptions, spellings and readings. All the material is encoded in SGML and it includes over one million hypertext links which were generated by a computer program. This means that the investment in the project's data is carried forward from one delivery system to another, indefinitely into the future.

9. Making SGML Work Effectively

Getting started with SGML can seem to be a big hurdle to overcome, but in fact the actual mechanics of working with SGML are nowhere near as difficult as is often assumed. SGML tags are rarely typed in, but are normally inserted by software programs. WordPerfect 6.1 and 7 includes an SGML component and many projects use SoftQuad's Author/Editor for data entry. These programs can incorporate a template which is filled in with data. Like other SGML software they make use of the DTD. They know which tags are valid at any position in the document and can offer only those to the user who can pick from a menu. They can also provide a pick list of attributes and their values if these are a closed set. They ensure that what is produced is a valid SGML document. They can also toggle the display of tags on and off very easily - Author/Editor and other SoftQuad products enclose them in boxes which are very easy to see. They also incorporate style sheets which define the display format for every element.

Nevertheless, inserting tags in this way can be rather cumbersome and various software tools exist to help in the translation of "legacy" data to SGML. Of course, these tools cannot add intelligence to data if it was not there in the legacy format, but they can do a reasonable and lowcost job of converting material for large scale projects where only broad structural information is needed. For those who are familiar with UNIX, the shareware program sgmls and its successor sp are excellent tools for validating SGML documents and can be incorporated in processing programs. There are also ways in which the markup can be minimized. End tags can be omitted in some circumstances, for example in a list where the start of a new list item implies that the previous one has ended.

There is no doubt that SGML is considered expensive by some projects, but the pay-off can be seen many times over further down the line. The quick and dirty solution to a computing problem does not last very long and history has shown how much time can be wasted converting from one system to another or how much data can be lost because it is in a proprietary system. It is rather surprising that the simple notion of encoding what the parts of a document are, rather than what the computer is supposed to do with them, took so long to catch on. Much of the investment in any computer project is in the data and SGML is the best way we know so far of ensuring that the data will last for a long time and that it can be used and re-used for many different purposes. It also ensures that the project is not dependent on one software vendor. Projects are always under pressure to produce results and this can be done simply with SGML documents by using SoftQuad's Panorama SGML viewer.[15] Panorama immediately gives a sense of what is possible and is easy to use.

The amount of encoding is obviously a key factor in the cost and so any discussion about the

cost-effectiveness of an SGML project should really always be made with reference to the specific DTD in use and the level of markup to be inserted. (Unfortunately at present this seems to be rarely the case.) It is quite possible, although clearly not sensible, to have a valid SGML document which consists of one start tag at the beginning and one at the end with no other markup in between. At the other extreme each word (or even letter) in the document could have several layers of markup attached to it. What is clear is that the more markup there is, the more useful the document is and the more expensive it is to create. As far as I am aware, little research has been done on the optimum level of markup, but at least with SGML it is possible to add markup to a document later without prejudicing what is already encoded.

SGML does have one fairly significant weakness. It assumes that each document is a single hierarchic structure, but in the real world (at least of the humanities) very few documents are as simple as this.[16] For example, a printed edition of a play has one structure of acts, scenes and speeches and another of pages and line numbers. A new act or scene does not normally start on a new page and so there is no relationship between the pages and the act and scene structure. It is simply an accident of the typography. The problem arises even with paragraphs in prose texts, since a new page does not start with a new paragraph, or a new paragraph with a new page. For well-known editions the page numbers are important, but they cannot easily be encoded in SGML other than as "empty" tags which simply indicate a point in the text, not the beginning and end of a piece of information. The disadvantage here is that the processing of information marked by empty tags cannot make full use of SGML's capabilities. Another example of the same problem is quotations spanning over paragraphs. They have to be closed and then opened again with attributes to indicate that they are really all the same quotation.

For many scholars, SGML is exciting to work with because it opens up so many more possibilities for working with source material. We now have a much better way than ever before of representing in electronic form the kinds of interpretation and discussion which are the basis of scholarship in the humanities. But as we begin to understand this, some new challenges appear.[17] What happens when documents from different sources (and thus different DTDs) are merged into the same database? In theory, computers make it very easy to do this, but how do we merge material that has been encoded according to different theoretical perspectives and retain the identification and individuality of each perspective? It is possible to build some kind of "mega-DTD", but this may become so free in structure that it is difficult to do any useful processing of the material.

Attention must now turn to making SGML work more effectively. Finding better ways of adding markup to documents is a high priority. The tagging could be speeded up by a program which can make intelligent guesses for the tagging based on information it has derived from similar material that has already been tagged, much in the same way as some word class tagging programs "learn" from text that has already been tagged manually. We also need to find ways of linking encoded text to digital images of the same material without the need for hand-coding. Easier ways must be found for handling multiple parallel structures. All research leading to better use of SGML could benefit from a detailed analysis of documents that have already been encoded in SGML. The very fact that they are in SGML makes this easy to do.

NOTES:

[1] Ian Graham's *HTML Sourcebook: a complete guide to HTML 3.0*, 2nd edition, Wiley, 1996, especially the beginning of Chapter 3, gives an excellent overview of the characteristics of a book in the context of a discussion of the design of electronic resources. The third edition of this book was published early in 1997.

[2] Jay David Bolter's *Writing Spaces: the computer, hypertext and the history of writing*, Erlbaum, 1991, expands on some of these ideas. See also George Landow, *Hypertext: the convergence of contemporary critical theory and technology*, Johns Hopkins, 1992, and my own *Knowledge Representation*, a paper commissioned as part of the Getty Art History Information Program (now the Getty Information Institute) Research Agenda for Humanities Computing, published in *Research Agenda for Networked Cultural Heritage*, p. 31-34, Getty Information Institute, and also available at http://www.ahip.getty.edu/agenda/represen.html.

[3] These terms have been used, among others, by the Model Editions Partnership (http://mep.cla.sc.edu).

[4] This was the planning meeting for the Text Encoding Initiative project. It was held in November 1987.

[5] C.J. Date, *An Introduction to Database Systems*, 4th edition, Addison Wesley, 1986 is a good introduction to relational database technology.

[6] By far the most useful starting point for information about SGML is the very comprehensive Web site at http://www.sil.org/sgml/. This is maintained and updated very regularly by Robin Cover of the Summer Institute for Linguistics.

[7] The TEI's Web site is at http://www.uic.edu/orgs/tei/. It contains links to electronic versions of the TEI Guidelines and DTDs as well as projects which are using the DTD.

[8] See Richard Giordano, "The Documentation of Electronic Texts Using Text Encoding Initiative Headers: an Introduction", *Library Resources and Technical Services*, 38 (1994), 389ff for a detailed discussion of the header from the perspective of someone who is both a librarian and a computer scientist.

[9] More information about the EAD can be found at http://lcweb.loc.gov/ead/. This site has examples of the Library of Congress EAD projects. Others can be found via links from the SGML Web site.

[10] This example can be seen at http://www.ceth.rutgers.edu/projects/griffis/project.htm. The site also provides instructions for downloading the Panorama SGML viewer.

[11] See Yuri Rubinsky, "Electronic Texts the Day After Tomorrow", p5-13 in *Visions and Opportunities in Electronic Publishing: Proceedings of the Second Symposium, December 5-8, 1992*, edited by Ann Okerson, Association for Research Libraries, also available at http://arl.cni.org:80/scomm/symp2/Rubinsky.html. Rubinsky was the founder of SoftQuad and a

leading figure in the SGML community until his tragic early death in January 1996.

[12] There is a very useful set of Frequently Asked Questions (FAQ) on XML at http://www.ucc.ie/xml/. See also the XML section of the SGML Web site at http://www.sil.org/sgml/related.html#xml.

[13] See note (3).

[14] The Orlando Project's Web site is at http://www.ualberta.ca/ORLANDO/.

[15] A free version of Panorama can be used as Web helper application. The Professional version runs as a standalone program. It is well within the price range of an academic user and, together with WordPerfect 7, provides a cheap way of beginning to work with SGML.

[16] In order to deal with the problem of overlap, the Wittgenstein Archives at the University of Bergen (http://www.hd.uib.no/wab/) have devised their own encoding scheme MECS (Multi-Element Code System). MECS contains some of the properties of SGML, but has simpler mechanisms for structures which are cumbersome in SGML. However this has meant that they have had to develop their own software to process the material.

[17] For a longer discussion of new questions posed by the use of SGML and especially its perceived lack of semantics, see C.M. Sperberg-McQueen's closing address to the SGML92 conference at http://www.sil.org/sgml/sgml92sp.html. He notes: `In identifying some areas as promising new results, and inviting more work, there is always the danger of shifting from "inviting more work" to "needing more work" and giving the impression of dissatisfaction with the work that has been accomplished. I want to avoid giving that impression, because it is not true, so I want to make very clear: the questions I am posing are not criticisms of SGML. On the contrary, they are its children. .... SGML has created the environment within which these problems can be posed for the first time, and I think part of its accomplishment is that by solving one set of problems, it has exposed a whole new set of problems.'

. . . . . .

For additional information about the conference, or The Andrew W. Mellon Foundation's scholarly communication initiatives, please contact Richard Ekman. For additional information about ARL or this web site contact Patricia Brennan, ARL Program Officer at (202) 296-2296.

Return to Office of Scholarly Communication Home Page

ARL Home ARL Scholarly Communication and Technology Home Page

© Association of Research Libraries, Washington, DC
Web Design by Angelo F. Cruz